

Estimating Probabilities in Recommendation Systems

Mingxuan Sun* Guy Lebanon

Georgia Institute of Technology
Atlanta GA 30332 USA

Paul Kidwell

Lawrence Livermore National Lab
Livermore CA 94550 USA

December 3, 2010

Abstract

Recommendation systems are emerging as an important business application with significant economic impact. Currently popular systems include Amazon’s book recommendations, Netflix’s movie recommendations, and Pandora’s music recommendations. In this paper we address the problem of estimating probabilities associated with recommendation system data using non-parametric kernel smoothing. In our estimation we interpret missing items as randomly censored observations and obtain efficient computation schemes using combinatorial properties of generating functions. We demonstrate our approach with several case studies involving real world movie recommendation data. The results are comparable with state-of-the-art techniques while also providing probabilistic preference estimates outside the scope of traditional recommender systems.

1 Introduction

Recommendation systems are emerging as an important business application with significant economic impact. The data in such systems are collections of incomplete tied preferences across n items associated with m different users. Given an incomplete tied preference associated with an additional $m + 1$ user, the system recommends unobserved items to that user based on the preference relations of the $m + 1$ users. Currently deployed recommendation systems include book recommendations at amazon.com, movie recommendations at netflix.com, and music recommendations at pandora.com. Constructing accurate recommendation systems (that recommend to users items that are truly preferred over other items) is important for assisting users as well as increasing business profitability. It is an important unsolved goal in machine learning and data mining.

In most cases of practical interest the number of items n indexed by the system (items may be books, movies, songs, etc.) is relatively high in the $10^3 - 10^4$ range. Perhaps due the size of n , it is almost always the case that each user observes only a small subset of the items, typically in the range 10-100. As a result the preference relations expressed by the users are over a small subset of the n items.

Formally, we have m users providing incomplete tied preference relations on n items

$$\begin{aligned} S_1 : & \quad A_{1,1} \prec A_{1,2} \prec \cdots \prec A_{1,k(1)} \\ S_2 : & \quad A_{2,1} \prec A_{2,2} \prec \cdots \prec A_{2,k(2)} \\ & \quad \vdots \\ S_m : & \quad A_{m,1} \prec A_{m,2} \prec \cdots \prec A_{m,k(m)} \end{aligned} \tag{1}$$

where $A_{i,j} \subset \{1, \dots, n\}$ are sets of items (wlog we identify items with integers $1, \dots, n$) defined by the following interpretation: user i prefers all items in $A_{i,j}$ to all items in $A_{i,j+1}$. The notation $k(i)$ above is

*Corresponding author: msun3@gatech.edu

the number of such sets provided by user i . The data (1) is incomplete since not all items are necessarily observed by each user i.e., $\bigcup_{j=1}^{k(i)} A_{i,j} \subsetneq \{1, \dots, n\}$ and may contain ties since some items are left uncomparated, i.e., $|A_{i,j}| > 1$. Recommendation systems recommend items to a new user, denoted as $m+1$, based on their preference

$$S_{m+1} : A_{m+1,1} \prec A_{m+1,2} \prec \dots \prec A_{m+1,k(m+1)} \quad (2)$$

and its relation to the preferences of the m users (1).

As an illustrative example, assuming $n = 9, m = 3$, the data

$$\begin{aligned} S_1 : & \quad 1, 8, 9 \prec 4 \prec 2, 3, 7 \\ S_2 : & \quad 4 \prec 2, 3 \prec 8 \\ S_3 : & \quad 4, 8 \prec 2, 6, 9 \end{aligned}$$

corresponds to $A_{1,1} = \{1, 8, 9\}$, $A_{1,2} = \{4\}$, $A_{1,3} = \{2, 3, 7\}$, $A_{2,1} = \{4\}$, $A_{2,2} = \{2, 3\}$, $A_{2,3} = \{8\}$, $A_{3,1} = \{4, 8\}$, $A_{3,2} = \{2, 6, 9\}$, and $k(1) = k(2) = 3, k(3) = 2$. From the data we may guess that item 4 is relatively popular across the board while some users like item 8 (users 1, 3) and some hate it (user 2). Given a new $m+1$ user issuing the preference $1 \prec 2, 3, 7$ we might observe a similar pattern of preference or taste as user 1 and recommend to the user item 8. We may also recommend item 4 which has broad appeal resulting in the augmentation

$$1 \prec 2, 3, 7 \quad \mapsto \quad 1, 4, 8 \prec 2, 3, 7.$$

We note that in some cases the preference relations (1) arise from users providing numeric scores to items. For example, if the users assign 1-5 stars to movies, the set $A_{i,j}$ contains all movies that user i assigned $6-j$ stars to and $k(i) = 5$ (assuming some movies were assigned to each of the 1, 2, 3, 4, 5 star levels). As pointed out by a wide variety of studies in economics and social sciences, such numeric scores are inconsistent among different users. We therefore proceed to interpret such data as ordinal rather than numeric.

A substantial body of literature in computer science has addressed the problem of constructing recommendation systems. We have attempted to outline the most important and successful approaches in the related work section towards the end of this paper. However, none of these previous approaches are fully satisfactory from a statistical perspective: there are no reasonable probability models assumed to generate the data and no clear meaningful statistical estimation procedures. We substantiate this argument more fully in the related work section.

In this paper we describe a non-parametric statistical technique for estimating probabilities on preferences based on the data (1). This technique may be used in recommendation systems in different ways. Its principal usage may be to provide a statistically meaningful estimation framework for issuing recommendations (in conjunction with decision theory). However, it also leads to other important applications including mining association rules, exploratory data analysis, and clustering items and users. Two key observations that we make are: (i) incomplete tied preference data may be interpreted as randomly censored permutation data, and (ii) using generating functions we are able to provide a computationally efficient scheme for computing the estimator in the case of triangular smoothing.

We proceed in the next sections to describe notations and our assumptions and estimation procedure, and follow with case studies demonstrating our approach on real world recommendation systems data.

2 Definitions and Estimation Framework

We describe the following notations and conventions for permutations, which are taken from [3] where more detail may be found. We denote a permutation by listing the items from most preferred to least separated by a \prec or $|$ symbol: $\pi^{-1}(1) \prec \pi^{-1}(2) \prec \dots \prec \pi^{-1}(n)$, e.g. $\pi(1) = 2, \pi(2) = 3, \pi(3) = 1$ is $3 \prec 1 \prec 2$. Ranking with ties occur when judges do not provide enough information to construct a total order. In particular, we define tied rankings as a partition of $\{1, \dots, n\}$ to $k < n$ disjoint subsets $A_1, \dots, A_k \subset \{1, \dots, n\}$ such that all items in A_i are preferred to all items in A_{i+1} but no information is provided concerning the relative

preference of the items among the sets A_i . We denote such rankings by separating the items in A_i and A_{i+1} with a \prec or $|$ notation. For example, the tied ranking $A_1 = \{3\}, A_2 = \{2\}, A_3 = \{1, 4\}$ (items 1 and 4 are tied for last place) is denoted as $3 \prec 2 \prec 1, 4$ or $3|2|1, 4$.

Ranking with missing items occur when judges omit certain items from their preference information altogether. For example assuming a set of items $\{1, \dots, 4\}$, a judge may report a preference $3 \prec 2 \prec 4$, omitting altogether item 1 which the judge did not observe or experience. This case is very common in situations involving a large number of items n . In this case judges typically provide preference only for the $l \ll n$ items that they observed or experienced. For example, in movie recommendation systems we may have $n \sim 10^3$ and $l \sim 10^1$.

Rankings can be full (permutations), with ties, with missing items, or with both ties and missing items. In either case we denote the rankings using the \prec or $|$ notation or using the disjoint sets A_1, \dots, A_k notation. We also represent tied and incomplete rankings by the set of permutations that are consistent with it. For example,

$$\begin{aligned} 3 \prec 2 \prec 1, 4 &= \{3 \prec 2 \prec 1 \prec 4\} \cup \{3 \prec 2 \prec 4 \prec 1\} \\ 3 \prec 2 \prec 4 &= \{1 \prec 3 \prec 2 \prec 4\} \cup \{3 \prec 1 \prec 2 \prec 4\} \cup \{3 \prec 2 \prec 1 \prec 4\} \cup \{3 \prec 2 \prec 4 \prec 1\} \end{aligned}$$

are sets of two and four permutations corresponding to tied and incomplete rankings, respectively.

It is hard to directly posit a coherent probabilistic model on incomplete tied data such as (1). Different preferences relations are not unrelated to each other: they may subsume one another (for example $1 \prec 2 \prec 3$ and $1 \prec 3$), represent disjoint events (for example $1 \prec 3$ and $3 \prec 1$), or interact in more complex ways (for example $1 \prec 2 \prec 3$ and $1 \prec 4 \prec 3$). A valid probabilistic framework needs to respect the constraints resulting from the axioms of probability, e.g., $p(1 \prec 2 \prec 3) \leq p(1 \prec 3)$.

Our approach is to consider the incomplete tied preferences as censored permutations. That is, we assume a distribution $p(\pi)$ over permutations $\pi \in \mathfrak{S}_n$ (\mathfrak{S}_n is the symmetric group of permutations of order n) that describes the complete without-ties preferences in the population. The data available to the recommender system (1) is sampled by drawing m iid permutations from p : $\pi_1, \dots, \pi_m \stackrel{\text{iid}}{\sim} p$, followed by censoring to result in the observed preferences S_1, \dots, S_m

$$\pi_i \sim p(\pi), \quad S_i \sim p(S|\pi_i), \quad i = 1, \dots, m+1 \quad (3)$$

$$p(\pi|S) = \frac{I(\pi \in S)p(\pi)}{\sum_{\sigma \in S} p(\sigma)} \quad (4)$$

$$p(S|\pi) = p(\pi|S)p(S)/p(\pi) = \frac{I(\pi \in S)p(\pi)p(S)}{p(\pi)\sum_{\sigma \in S} p(\sigma)} = \frac{I(\pi \in S)p(S)}{\sum_{\sigma \in S} p(\sigma)} \quad (5)$$

where $p(S)$ is the probability of observing the censoring S (specifically, it is not equal to $\sum_{\sigma \in S} p(\sigma)$).

Although many approaches for estimating p given S_1, \dots, S_m are possible, experimental evidence point to the fact that in recommendation systems with high n , the distribution p does not follow a simple parametric form such as the Mallows, Bradley-Terry, or Thurstone models [12] (see Figure 1 for a demonstration how parametric assumptions break down with increasing n). Instead, the distribution p tends to be diffuse and multimodal with different probability mass regions corresponding to different types of judges (for example in movie preferences probability modes may correspond to genre as fans of drama, action, comedy, etc. having similar preferences).

We therefore propose to estimate the underlying distribution p on permutations using non-parametric kernel smoothing. The standard kernel smoothing formula applies to the permutation setting as

$$\hat{p}(\pi) = \frac{1}{m} \sum_{i=1}^m K_h(T(\pi, \pi_i))$$

where $\pi_1, \dots, \pi_m \stackrel{\text{iid}}{\sim} p$, T a distance on permutations such as Kendall's distance and $K_h(r) = h^{-1}K(r/h)$ a normalized unimodal function. In the case at hand, however, the observed preferences π_i as well as π are

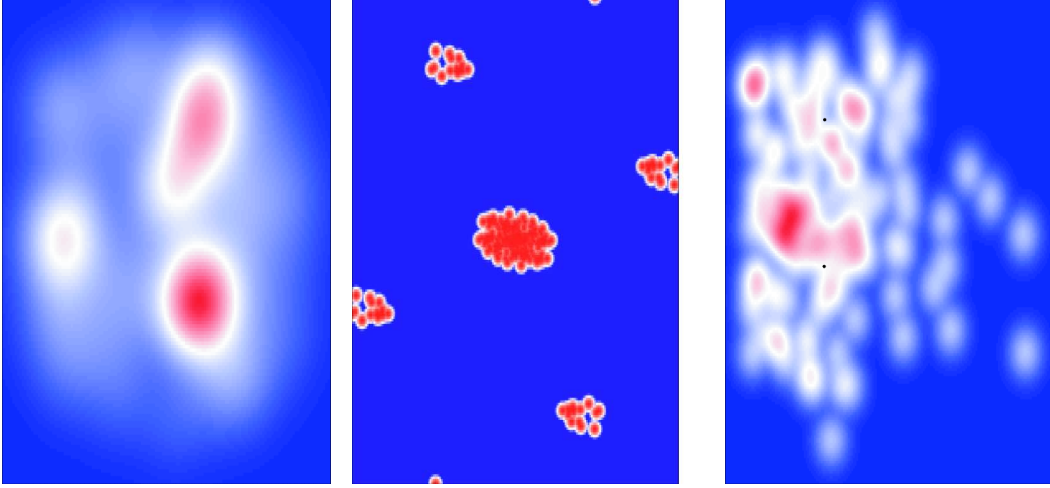


Figure 1: Heat map visualization of the density of ranked data using multidimensional scaling with expected Kendall's Tau distance. The datasets are APA voting (left, $n = 5$), Jester (middle, $n = 100$), and EachMovie (right, $n = 1628$) datasets. None of these cases show a simple parametric form, and the complexity of the density increases with the number of items n . This motivates the use of non-parametric estimators for modeling preferences over a large number of items.

replaced with permutations sets S_1, \dots, S_m, R representing incomplete tied preferences

$$\hat{p}(R) = \sum_{\pi \in R} \hat{p}(\pi) = \frac{1}{m} \sum_{i=1}^m \sum_{\pi \in R} \sum_{\sigma \in S_i} q(\sigma|S_i) K_h(T(\pi, \sigma)) \quad (6)$$

where $q(\sigma|S_i)$ serves as a surrogate for the unknown $p(\sigma|S_i) \propto I(\sigma \in S_i)p(\sigma)$ (see (4)). Selecting $q(\sigma|S_i) = p(\sigma|S_i)$ would lead to consistent estimation of $p(R)$ in the limit $h \rightarrow 0$, $m \rightarrow \infty$ assuming positive $p(\pi), p(S)$. Such a selection, however, is generally impossible since $p(\pi)$ and therefore $p(\sigma|S_i)$ are unknown.

In general the specific choice of the surrogate $q(\sigma|S)$ is important as it may influence the estimated probabilities. Furthermore, it may cause underestimation or overestimation of $\hat{p}(R)$ in the limit of large data. An exception occurs when the sets S_1, \dots, S_m are either subsets of R or disjoint from R . In this case $\lim_{h \rightarrow 0} K_h(\pi, \sigma) = I(\pi = \sigma)$ resulting in the following limit (with probability 1 by the strong law of large numbers)

$$\lim_{m \rightarrow \infty} \lim_{h \rightarrow 0} \hat{p}(R) = \lim_{m \rightarrow \infty} \frac{1}{m} \sum_{i=1}^m I(S_i \subset R) \sum_{\sigma \in S_i} q(\sigma|S_i) \quad (7)$$

$$= \lim_{m \rightarrow \infty} \frac{1}{m} \sum_{i=1}^m I(S_i \subset R) = \lim_{m \rightarrow \infty} \frac{1}{m} \sum_{i=1}^m I(\pi_i \in R) = p(R). \quad (8)$$

Thus, if we our data is comprised of preferences S_i that are either disjoint or a subset of R we have consistency *regardless* of the choice of the surrogate q . Such a situation is more realistic when the preference R involves a small number of items and the preferences S_i , $i = 1, \dots, m$ involve a larger number of items. This is often the case for recommendation systems where individuals report preferences over 10-100 items and we are mostly interested in estimating probabilities of preferences over fewer items such as $i \prec j, k$ or $i \prec j, k \prec l$ (see experiment section).

The main difficulty with the estimator above is the computation of $\sum_{\pi \in R} \sum_{\sigma \in S_i} q(\sigma|S_i) K_h(T(\pi, \sigma))$. In the case of high n and only a few observed items k the sets S_i, R grow factorially as $(n - k)!$ making

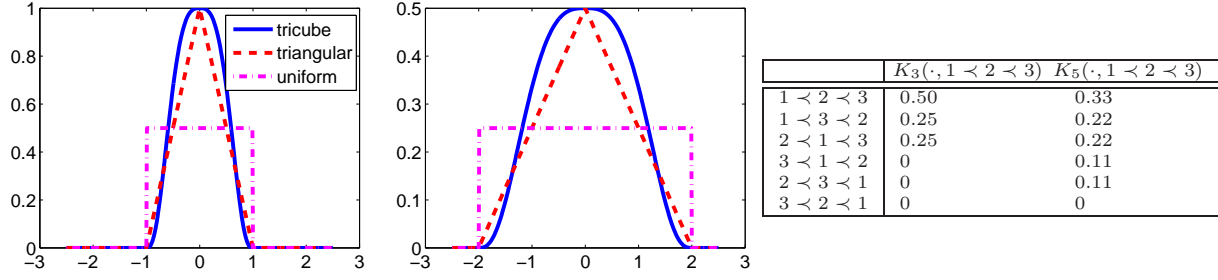


Figure 2: Tricube, triangular, and uniform kernels on \mathbb{R} with bandwidth $h = 1$ (left) and $h = 2$ (middle). Right: triangular kernel on permutations ($n = 3$).

a naive computation of (6) intractable for all but the smallest n . In the next section we explore efficient computations of these sums for a triangular kernel K_h and a uniform $q(\pi|S)$.

3 Computationally Efficient Kernel Smoothing

In previous work [10] the estimator (6) is proposed for tied (but complete) rankings. That work derives closed form expressions and efficient computation for (6) assuming a Mallows kernel [11]

$$K_h(T(\pi, \sigma)) = \exp\left(-\frac{T(\pi, \sigma)}{h}\right) \prod_{j=1}^n \frac{1 - e^{-1/h}}{1 - e^{-j/h}} \quad (9)$$

where T is Kendall's Tau distance on permutations (below $I(x) = 1$ for $x > 0$ and 0 otherwise)

$$T(\pi, \sigma) = \sum_{i=1}^{n-1} \sum_{l>i} I(\pi\sigma^{-1}(i) - \pi\sigma^{-1}(l)). \quad (10)$$

Unfortunately these simplifications do not carry over to the case of incomplete rankings where the sets of consistent permutations S_1, \dots, S_m are not cosets of the symmetric group. As a result the problem of probability estimation in recommendation systems where n is high and many items are missing is particularly challenging. However, as we show below replacing the Mallows kernel (9) with a triangular kernel leads to efficient computation in some cases. Specifically, the triangular kernel on permutation is

$$K_h(T(\pi, \sigma)) = (1 - h^{-1}T(\pi, \sigma)) I(h - T(\pi, \sigma)) / C \quad (11)$$

where the bandwidth parameter h represent both the support (the kernel is 0 for all larger distances) and the inverse slope of the triangle. As we show below the normalization term C is a function of h and may be efficiently computed using generating functions. Figure 2 (right panel) displays the linear decay of (11) for the simple case of permutations over $n = 3$ items.

Combinatorial Generating Function

Generating functions, a tool from enumerative combinatorics, allow efficient computation of (6) by concisely expressing the distribution of distances between permutations. Kendall's tau $T(\pi, \sigma)$ is the total number of discordant pairs or inversions between π, σ [20] and thus its computation becomes a combinatorial counting problem. We associate the following generating function with the symmetric group of order n permutations

$$G_n(z) = \prod_{j=1}^{n-1} \sum_{k=0}^j z^k. \quad (12)$$

As shown for example in [20] the coefficient of z^k of $G_n(z)$, which we denote as $[z^k]G_n(z)$, corresponds to the number of permutations σ for which $T(\sigma, \pi') = k$. For example, the distribution of Kendall's tau $T(\cdot, \pi')$ over all permutations of 3 items is described by $G_3(z) = (1+z)(1+z+z^2) = 1z^0 + 2z^1 + 2z^2 + 1z^3$ i.e., there is one permutation σ with $T(\sigma, \pi') = 0$, two permutations σ with $T(\sigma, \pi') = 1$, two with $T(\sigma, \pi') = 2$ and one with $T(\sigma, \pi') = 3$. Another important generating function is

$$H_n(z) = \frac{G_n(z)}{1-z} = (1+z+z^2+z^3+\dots)G_n(z)$$

where $[z^k]H_n(z)$ represents the number of permutations σ for which $T(\sigma, \pi') \leq k$.

Proposition 1. *The normalization term $C(h)$ is given by $C(h) = [z^h]H_n(z) - h^{-1}[z^{h-1}]\frac{G'_n(z)}{1-z}$.*

Proof. The proof factors the non-normalized triangular kernel $CK_h(\pi, \sigma)$ to $I(h-T(\pi, \sigma))$ and $h^{-1}T(\pi, \sigma)I(h-T(\pi, \sigma))$ and making the following observations. First we note that summing the first factor over all permutations may be counted by $[z^h]H_n(z)$. The second observation is that $[z^{k-1}]G'_n(z)$ is the number of permutations σ for which $T(\sigma, \pi') = k$, multiplied by k . Since we want to sum over that quantity for all permutations whose distance is less than h we extract the $h-1$ coefficient of the generating function $G'_n(z) \sum_{k \geq 0} z^k = G'_n(z)/(1-z)$. We thus have

$$C = \sum_{\sigma: T(\pi', \sigma) \leq h} 1 - h^{-1} \sum_{\sigma: T(\pi', \sigma) \leq h} T(\pi', \sigma) = [z^h]H_n(z) - h^{-1}[z^{h-1}]\frac{G'_n(z)}{1-z}.$$

□

Proposition 2. *The complexity of computing $C(h)$ is $O(n^4)$.*

Proof. We describe a dynamic programming algorithm to compute the coefficients of G_n by recursively computing the coefficients of G_k from the coefficients of G_{k-1} , $k = 1, \dots, n$. The generating function $G_k(z)$ has $k(k+1)/2$ non-zero coefficients and computing each of them (using the coefficients of G_{k-1}) takes $O(k)$. We thus have $O(k^3)$ to compute G_k from G_{k-1} which implies $O(n^4)$ to compute G_k , $n = 1, \dots, n$. We conclude the proof by noting that once the coefficients of G_n are computed the coefficients of $H_n(z)$ and $G_n(z)/(1-z)$ are computable in $O(n^2)$ as these are simply cumulative weighted sums of the coefficients of G_n . □

Note that computing $C(h)$ for one or many h values may be done offline prior to the arrival of the rankings and the need to compute the estimated probabilities.

Denoting by k the number of items ranked in either S or R or both, the computation of $\hat{p}(\pi)$ in (6) requires $O(k^2)$ online and $O(n^4)$ offline complexity if either non-zero smoothing is performed over the entire data i.e., $\max_{\pi \in R} \max_{i=1}^n \max_{\sigma \in S_i} T(\sigma, \pi) < h$ or alternatively, we use the modified triangular kernel $K_h^*(\pi, \sigma) \propto (1 - h^{-1})T(\pi, \sigma)$ which is allowed to take negative values for the most distant permutations (normalization still applies though).

Proposition 3. *For two sets of permutations S, R corresponding to tied-incomplete rankings*

$$\frac{1}{|S||R|} \sum_{\pi \in S} \sum_{\sigma \in R} T(\pi, \sigma) = \frac{n(n-1)}{4} - \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=i+1}^n (1 - 2p_{ij}(S))(1 - 2p_{ij}(R)) \quad (13)$$

$$p_{ij}(U) = \begin{cases} I(\tau_U(j) - \tau_U(i)) & i \text{ and } j \text{ are ranked in } U \text{ with } \tau_U(i) \neq \tau_U(j) \\ 1 - \frac{\tau_U(i) + \frac{\phi_U(i)-1}{2}}{k+1} & \text{only } i \text{ is ranked in } U \\ \frac{\tau_U(j) + \frac{\phi_U(j)-1}{2}}{k+1} & \text{only } j \text{ is ranked in } U \\ 1/2 & \text{otherwise} \end{cases}.$$

with $\tau_U(i) = \min_{\pi \in U} \pi(i)$, and $\phi_U(i)$ being the number of items that are tied to i in U .

Proof. We note that (13) is an expectation with respect to the uniform measure. We thus start by computing the probability $p_{ij}(U)$ that i is preferred to j for $U = S$ and $U = R$ under the uniform measure. Five scenarios exist for each of $p_{ij}(U)$ corresponding to whether each of i and j are ranked by S, R . Starting with the case that i is not ranked and j is ranked, we note that i is equally likely to be preferred to any item or to be preferred to. Given the uniform distribution over compatible rankings item j is equally likely to appear in positions $\tau_U(j), \dots, \tau_U(j) + \phi_U(j) - 1$. Thus

$$p_{ij} = \frac{1}{\phi_U(j)} \frac{\tau_U(j)}{k+1} + \dots + \frac{1}{\phi_U(j)} \frac{\tau_U(j) + \phi_U(j) - 1}{k+1} = \frac{\tau_U(j) + \frac{\phi_U(j)-1}{2}}{k+1} \quad (14)$$

Similarly, if j is unknown and i is known then $p_{ij} + p_{ji} = 1$. If both i and j are unknown either ordering must be equally likely given the uniform distribution making $p_{ij} = 1/2$. Finally, if both i and j are known $p_{ij} = 1, 1/2, 0$ depending on their preference. Given p_{ij} , linearity of expectation, and the independence between rankings, the change in the expected number of inversions relative to the uniform expectation $n(n-1)/4$ can be found by considering each pair separately,

$$\begin{aligned} \text{ET}(i, j) &= \frac{1}{2}P(i \text{ and } j \text{ disagree}) - \frac{1}{2}P(i \text{ and } j \text{ agree}) \\ &= \frac{1}{2}(p_{ij}(\sigma)(1 - p_{ij}(\pi)) + (1 - p_{ij}(\sigma))p_{ij}(\pi)) - p_{ij}(\sigma)p_{ij}(\pi) - (1 - p_{ij}(\sigma))(1 - p_{ij}(\pi)) \\ &= \frac{-1}{2}(1 - 2p_{ij}(\sigma))(1 - 2p_{ij}(\pi)). \end{aligned}$$

Summing the $n(n-1)/2$ components yields the desired quantity. \square

Corollary 1. Denoting the number of items ranked by either S or R or both as k , and assuming either $h > \max_{\pi \in R} \max_{i=1}^n \max_{\sigma \in S_i} T(\sigma, \pi)$ or that the modified triangular kernel $K_h^*(\pi, \sigma) \propto (1 - h^{-1})T(\pi, \sigma)$ is used, the complexity of computing $\hat{p}(R)$ in (6) (assuming uniform $q(\pi|S_i)$) is $O(k^2)$ online and $O(n^4)$ offline.

Proof. The proof follows from noting that (6) reduces to $O(n^4)$ offline computation of the normalization term and $O(k^2)$ online computation of the form (13). \square

4 Applications and Case Studies

We divide our experimental study to three parts. In the first we examine the task of predicting probabilities. The remaining two parts use these probabilities for rank prediction and rule discovery.

In our experiments we used three datasets. The MovieLens dataset¹ contains one million ratings from 6040 users over 3952 movies. The EachMovie dataset² contains 2.6 million ratings from 74424 users over 1648 movies. The Netflix dataset³ contains 100 million movie ratings from 480189 users on 17770. In all of these datasets users typically rated only a small number of items. Histograms of the distribution of the number of votes per user, number of votes per item, and vote distribution appear in Figure 3.

4.1 Estimating Probabilities

We consider here the task of estimating $\hat{p}(R)$ where R is a set of permutations corresponding to a tied incomplete ranking. Such estimates may be used to compute conditional estimates $\hat{P}(R|S_{m+1})$ which are used to predict which augmentations R of S_{m+1} are highly probable. For example, given an observed preference $3 \prec 2 \prec 5$ we may want to compute $\hat{p}(8 \prec 3 \prec 2 \prec 5 | 3 \prec 2 \prec 5) = \hat{p}(8 \prec 3 \prec 2 \prec 5) / \hat{p}(3 \prec 2 \prec 5)$ to see whether item 8 should be recommended to the user.

¹<http://www.grouplens.org>

² <http://www.grouplens.org/node/76>

³<http://www.netflixprize.com/community>

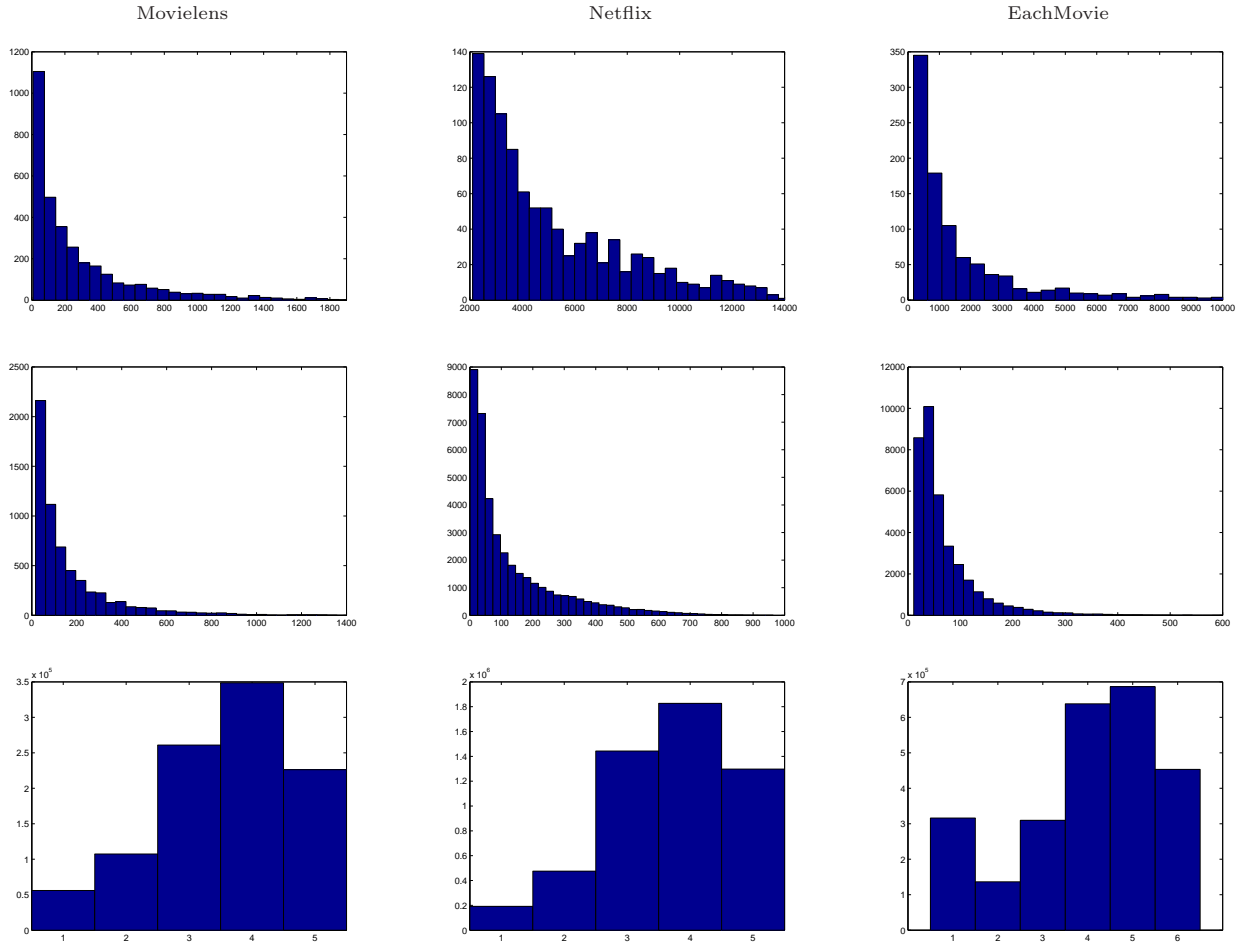


Figure 3: Histograms of the number of user votes per movie (top row), number of movies ranked per user (middle row), and votes (bottom row).

For simplicity we focus in this section on probabilities of simple events such as $i \prec j$ or $i \prec j \prec k$. The next section deals with more complex events. In our experiment, we estimate the probability of $i \prec j$ for the $n = 53$ most rated movies in Netflix and $m = 10000$ users who rate most of these movies. The probability matrix of the pairs is shown in Figure 4 where each cell corresponds to the probability of preference between a pair of movies determined by row j and column i . In the top left panel the rows and columns are ordered by average probability of a movie being preferred to others $r(i) = \frac{\sum_j \hat{p}(i \prec j)}{n}$ with the most preferred movie in row and column 1 (top right panel indicates the ordering according to $r(i)$). In the bottom left panel the movies were ordered first by popularity of genres and then by $r(i)$. The bottom right panel indicates that ordering. The names, genres, and both orderings of all 53 movies appear in Figure 6.

The three highest movies in terms of $r(i)$ are Lord of the Rings: The Return of the King, Finding Nemo, and Lord of the Rings: The Two Towers. The three lowest movies are Maid in Manhattan, Anger Management, and The Royal Tenenbaums. Examining the genre (colors in right panels of Figure 4) we see that family and science fiction are generally preferred to others movies while comedy and romance generally receive lower preferences. The drama, action genres are somewhere in the middle.

Also interesting is the variance of the movie preferences within specific genres. Family movies are generally preferred to almost all other movies. Science fiction movies, on the other hand, enjoy high preference overall but exhibit a larger amount of variability as a few movies are among the least preferred. Similarly, the preference probabilities of action movies are widely spread with some movies being preferred to others and others being less preferred. More specifically (see bottom left panel of Figure 4) we see that the decay of $r(i)$ within genres is linear for family and romance and nonlinear for science fiction, action, drama, and comedy. In these last three genres there are a few really “bad” movies that are substantially lower than the rest of the curve. Figure 6 shows the full information including titles, genres and orderings of the 53 most popular movies in Netflix.

We plot the individual values of $\hat{p}(i \prec j)$ for three movies: Shrek (family), Catch Me If You Can (drama) and Napoleon Dynamite (comedy) (Figure 5). Comparing the three stem plots we observe that Shrek is preferred to almost all other movies, Napoleon Dynamite is less preferred than most other movies, and Catch Me If You Can is preferred to some other movies but less preferred than others. Also interesting is the linear increase of the stem plots for Catch Me If You Can and Napoleon Dynamite and the non-linear increase of the stem plot for Shrek. This is likely a result of the fact that for very popular movies there are only a few comparable movies with the rest being very likely to be less preferred movies ($\hat{p}(i \prec j)$ close to 1).

In a second experiment (see Figure 7) we compare the predictive behavior of the kernel smoothing estimator with that of a parametric model (Mallows model) and the empirical measure (frequency of event occurring in the m samples). We evaluate the predictive performance of a probability estimator by separating the data to two parts: a training set that is used to construct the estimator and a testing set used for evaluation via its loglikelihood. A higher test set loglikelihood indicates that the model assigns high probability to events that occurred. Mathematically, this corresponds to approximating the KL divergence between nature and the model. Since the Mallows model is intractable for large n we chose in this experiment small values of n : 3, 4, 5.

We observe that the kernel estimator consistently achieves higher test set loglikelihood than the Mallows model and the empirical measure. The former is due to the breakdown of parametric assumptions as indicated by Figure 1 (note that this happens even for n as low as 3). The latter is due to the superior statistical performance of the kernel estimator over the empirical measure.

4.2 Rank Prediction

Our task here is to predict ranking of new unseen items for users. We follow the standard procedure in collaborative filtering: the set of users is partitioned to two sets, a training set and a testing set. For each of the test set users we further split the observed items into two sets: one set used for estimating preferences (together with the preferences of the training set users) and the second set to evaluate the performance of the prediction [14]. Given a loss function $L(i, j)$ which measures the loss of predicting rank i when true rank is j (rank here refers to the number of sets of equivalent items that are more or less preferred

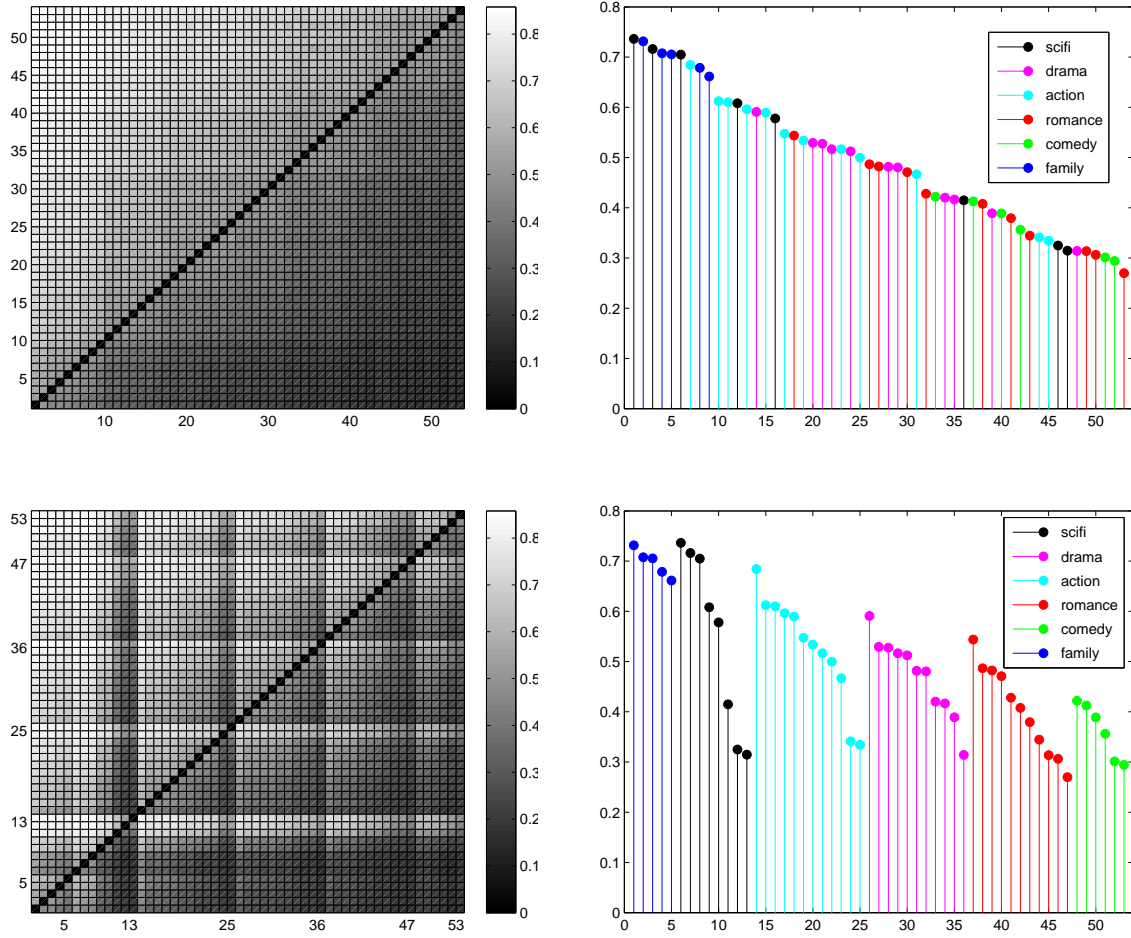


Figure 4: Left: The estimated probability of movie i being preferred to movie j . Right: a plot of $r(i) = \sum_j \hat{p}(i \prec j)/n$ for all movies with color indicating genres. In both panels the movies were ordered by $r(i)$ (top row) and first by popularity of genres and then by $r(i)$ (bottom row).

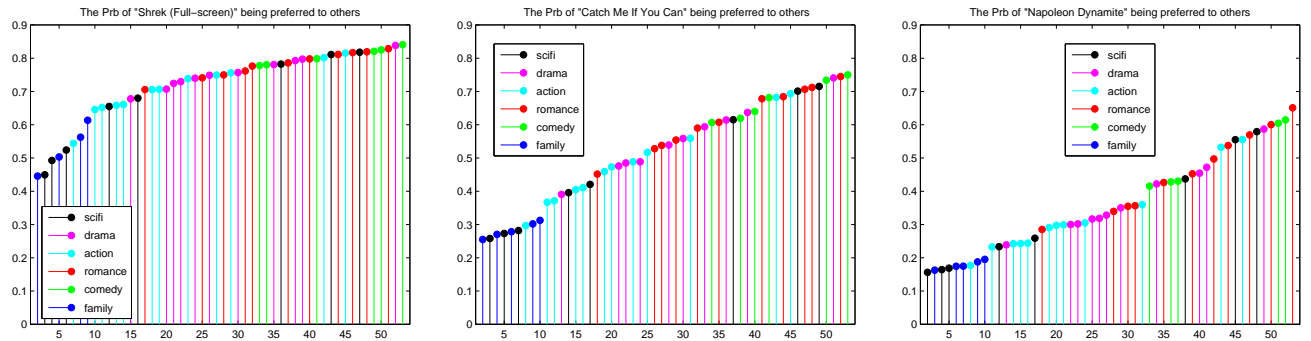


Figure 5: The value $\hat{p}(i \prec j)$ for all j for three movies: Shrek (left), Catch Me If You Can (middle) and Napoleon Dynamite (right).

Titles	Genre	Order1	Order2
Finding Nemo	6	2	1
Shrek	6	4	2
The Incredibles	6	5	3
Monsters, Inc.	6	8	4
Shrek II	6	9	5
LOTR: The Return of the King	1	1	6
LOTR: The Two Towers	1	3	7
LOTR: The Fellowship of the Ring	1	6	8
Spider-Man II	1	12	9
Spider-Man	1	16	10
The Day After Tomorrow	1	36	11
Tomb Raider	1	46	12
Men in Black II	1	47	13
Pirates of the Caribbean I	3	7	14
The Last Samurai	3	10	15
Man on Fire	3	11	16
The Bourne Identity	3	13	17
The Bourne Supremacy	3	15	18
National Treasure	3	17	19
The Italian Job	3	19	20
Kill Bill II	3	23	21
Kill Bill I	3	25	22
Minority Report	3	31	23
S.W.A.T.	3	44	24
The Fast and the Furious	3	45	25
Ocean's Eleven	2	14	26
I, Robot	2	20	27

Titles	Genre	Order1	Order2
Mystic River	2	21	28
Troy	2	22	29
Catch Me If You Can	2	24	30
Big Fish	2	28	31
Collateral	2	29	32
John Q	2	34	33
Pearl Harbor	2	35	34
Swordfish	2	39	35
Lost in Translation	2	48	36
50 First Dates	4	18	37
My Big Fat Greek Wedding	4	26	38
Something's Gotta Give	4	27	39
The Terminal	4	30	40
How to Lose a Guy in 10 Days	4	32	41
Sweet Home Alabama	4	38	42
Sideways	4	41	43
Two Weeks Notice	4	43	44
Mr. Deeds	4	49	45
The Wedding Planner	4	50	46
Maid in Manhattan	4	53	47
The School of Rock	5	33	48
Bruce Almighty	5	37	49
Dodgeball: A True Underdog Story	5	40	50
Napoleon Dynamite	5	42	51
The Royal Tenenbaums	5	51	52
Anger Management	5	52	53

Figure 6: The table contains the information of the 53 most popular movies of Netflix. Columns are movie titles, genres, order1 (the ordering in the upper row of Figure 4) and order2 (the ordering in the bottom row of Figure 4). Genres indicated by numbers from 1 to 6 represent science fiction, drama, action, romance, comedy, and family.

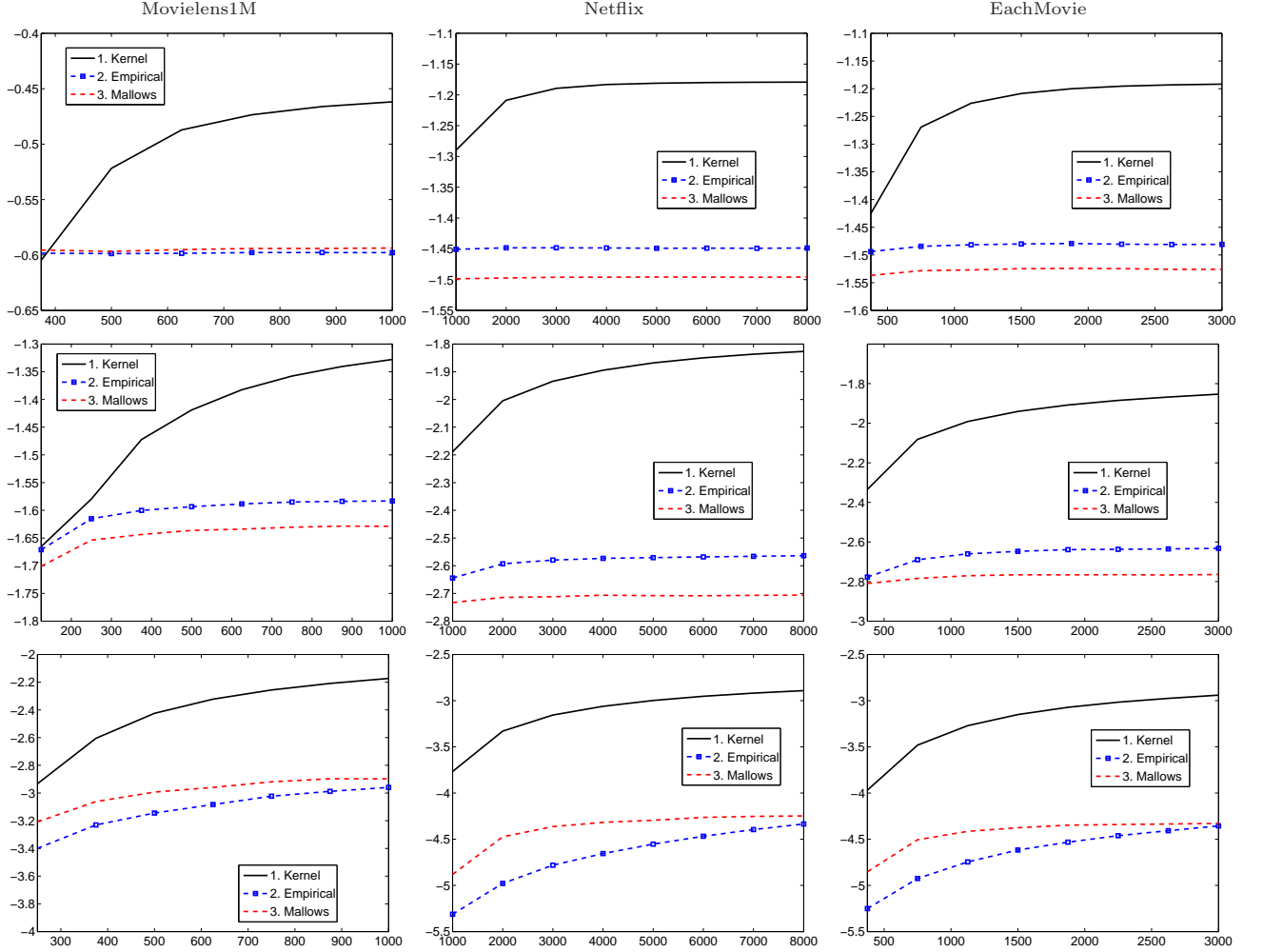


Figure 7: The test-set log-likelihood for kernel smoothing, Mallows model, and the empirical measure with respect to training size m for a small number of items $n = 3, 4, 5$ (top, middle, bottom rows) on three datasets. Both of the Mallows model (which is also intractable for large n which is why $n \leq 5$ in the experiment) and the empirical measure perform worse than the kernel estimator \hat{p} .

than the current item) we evaluate a prediction rule by the expected loss. We focus on three loss functions: $L_0(i, j) = 0$ if $i = j$ and 1 otherwise, $L_1(i, j) = |i - j|$ which reduces to the standard CF evaluation technique described in [14], and an asymmetric loss function (rows correspond to estimated number of stars (0-5) and columns to actual number of stars (0-5))

$$L_e = \begin{pmatrix} 0 & 0 & 0 & 3 & 4 & 5 \\ 0 & 0 & 0 & 2 & 3 & 4 \\ 0 & 0 & 0 & 1 & 2 & 3 \\ 9 & 4 & 1.5 & 0 & 0 & 0 \\ 12 & 6 & 3 & 0 & 0 & 0 \\ 15 & 8 & 4.5 & 0 & 0 & 0 \end{pmatrix}. \quad (15)$$

In contrast to the L_0 and L_1 loss, L_e captures the fact that recommending bad movies as good movies is worse than recommending good movies as bad.

For example, consider a test user whose observed preference is $3 \prec 4, 5, 6 \prec 10, 11, 12 \prec 23 \prec 40, 50, 60 \prec 100, 101$. We may withhold the preferences of items 4, 11 for evaluation purposes. The recommendation systems then predict a rank of 1 for item 4 and a rank of 4 for item 11. Since the true ranking of these items are 2 and 3 the absolute value loss is $|1 - 2| = 1$ and $|3 - 4| = 1$ respectively.

In our experiment, we use the kernel estimator \hat{p} to predict ranks that minimize the posterior loss and thus adapts to customized loss functions such as L_e . This is an advantage of a probabilistic modeling approach over more ad-hoc rule based recommendation systems.

Figure 8 compares the performance of our estimator to several standard baselines in the collaborative filtering literature: two older memory based methods vector similarity (sim1), correlation (sim2) e.g., [2], and a recent state-of-the-art non-negative matrix (NMF) factorization (gnmf) [9]. The kernel smoothing estimate performed similar to the state-of-the-art but substantially better than the memory based methods to which it is functionally similar.

4.3 Rule Discovery

In the third task, we used the estimator \hat{p} to detect noteworthy association rules of the type $i \prec j \Rightarrow k \prec l$ (if i is preferred to j than it is probably the case that k is preferred to l). Such association rules are important for both business analytics (devising marketing and manufacturing strategies) and recommendation system engineering. Specifically, we used \hat{p} to select sets of four items i, j, k, l for which the mutual information $I(i \prec j; k \prec l)$ is maximized. After these sets are identified we detected the precise shape of the rule (i.e., $i \prec j \Rightarrow k \prec l$ rather than $j \prec i \Rightarrow k \prec l$ by examining the summands in the mutual information expectation).

Figure 9 (top) shows the top 10 rules that were discovered. These rules nicely isolate viewer preferences for genres such as fantasy, romantic comedies, animation, and action (note however that genre information was not used in the rule discovery). To quantitatively evaluate the rule discovery process we judge a rule $i \prec j \Rightarrow k \prec l$ to be good if i, k are of the same genre and j, l are of the same genre. This quantitative evaluation appears in Figure 9 (bottom) where it is contrasted with the same rule discovery process (maximizing mutual information) based on the empirical measure.

In another rule discovery experiment, we used \hat{p} to detect association rules of the form i ranked highest $\Rightarrow j$ ranked second highest by selecting i, j that maximize the score $\frac{p(\pi(i)=1, \pi(j)=2)}{p(\pi(i)=1)p(\pi(j)=2)}$ between pairs of movies in the Netflix data. We similarly detected rules of the form i ranked highest $\Rightarrow j$ ranked lowest by maximizing the scores $\frac{p(\pi(i)=1, \pi(j)=\text{last})}{p(\pi(i)=1)p(\pi(j)=\text{last})}$ between pairs of movies.

The left panel of Figure 10 shows the top 9 rules of 100 most rated movies, which nicely represents movie preference of similar type, e.g. romance, comedies, and action. The right of Figure 10 shows the top 9 rules which represents like and dislike of different movie types, e.g. like of romance leads to dislike of action/thriller.

In a third experiment, we used \hat{p} to construct an undirected graph where vertices are items (Netflix movies) and two nodes i, j are connected by an edge if the average score of the rule i ranked highest \Rightarrow

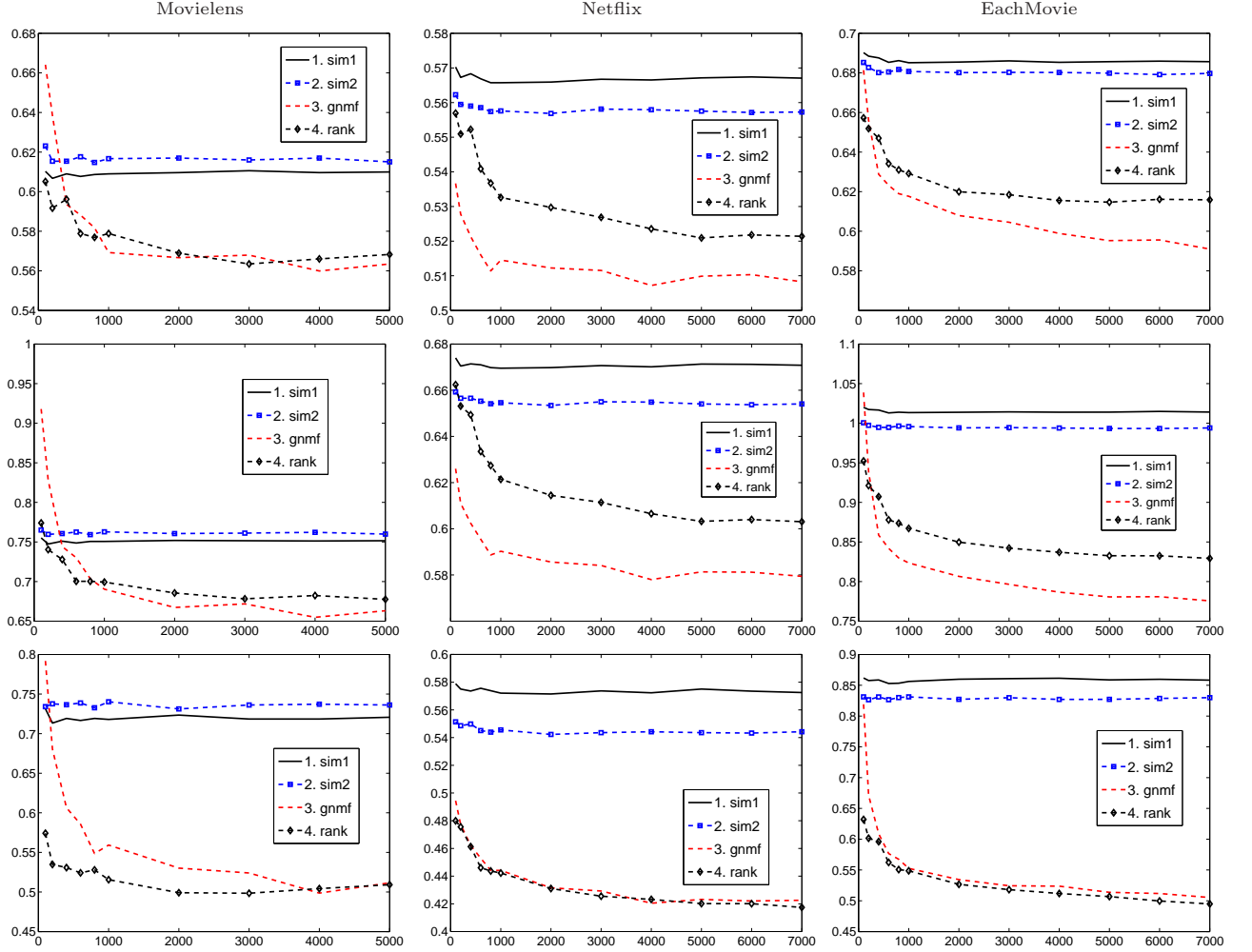


Figure 8: The prediction loss (top row: 0/1 loss L_0 , middle row: L_1 loss, bottom row: asymmetric loss L_e) with respect to training size on three datasets. The kernel smoothing estimate performed similar to the state-of-the-art gnmf (matrix factorization) but substantially better than the memory based methods to which it is functionally similar.

Shrek \prec LOTR: The Fellowship of the Ring	\Rightarrow	Shrek 2 \prec LOTR: The Return of the King
Shrek \prec LOTR: The Fellowship of the Ring	\Rightarrow	Shrek 2 \prec LOTR: The Two Towers
Shrek 2 \prec LOTR: The Fellowship of the Ring	\Rightarrow	Shrek \prec LOTR: The Return of the King
Kill Bill 2 \prec National Treasure	\Rightarrow	Kill Bill 1 \prec I. Robot
Shrek 2 \prec LOTR: The Fellowship of the Ring	\Rightarrow	Shrek 2 \prec LOTR: The Two Towers
LOTR: The Fellowship of the Ring \prec Monsters, Inc.	\Rightarrow	LOTR: The Two Towers \prec Shrek
National Treasure \prec Kill Bill 2	\Rightarrow	Pearl Harbor \prec Kill Bill 1
LOTR: The Fellowship of the Ring \prec Monsters, Inc.	\Rightarrow	LOTR: The Return of the King \prec Shrek
How to Lose a Guy in 10 Days \prec Kill Bill 2	\Rightarrow	50 First Dates \prec Kill Bill 1
I, Robot \prec Kill Bill 2	\Rightarrow	The Day After Tomorrow \prec Kill Bill 1

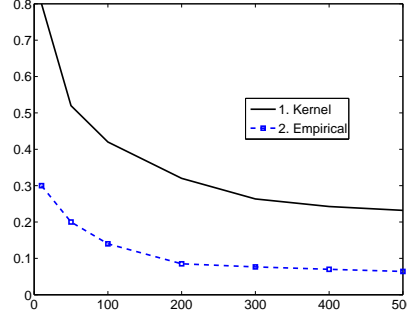


Figure 9: Top: top 10 rules discovered by the kernel smoothing estimator on Netflix in terms of maximizing mutual information. Bottom: a quantitative evaluation of the rule discovery. The x axis represents the number of rules discovered and the y axis represents the frequency of good rules in the discovered rules. Here a rule $i \prec j \Rightarrow k \prec l$ is considered good if i, k are of the same genre and j, l are of the same genre.

j ranked second highest and the rule j ranked highest $\Rightarrow i$ ranked second highest is higher than a certain threshold. Figure 11 shows the graph for the 100 most rated movies in Netflix (only movies with vertex degree greater than 0 are shown). The clusters in the graph corresponding to vertex color and numbering were obtained using a graph partitioning algorithm and the graph is embedded in a 2-D plane using standard graph visualization technique. Within each of the identified clusters movies are clearly similar with respect to genre, while an even finer separation can be observed when looking at specific clusters. For example, clusters 6 and 9 both contain comedy movies, where as cluster 6 tends toward slapstick humor and cluster 9 contains romantic comedies.

5 Related Work

Collaborative filtering or recommendation system has been an active research area in computer science since the 1990s. The earliest efforts made a prediction for the rating of items based on the similarity of the test user and the training users [17, 2, 6]. Specifically, these attempts used similarity measures such as Pearson correlation [17] and Vector cosine similarity [2, 6] to evaluate the similarity level between different users.

More recent work includes user and movie clustering [2, 21, 22], item-item similarities [18], Bayesian networks [2], dependence network [5] and probabilistic latent variable models [14, 7, 13].

Most recently, the state of the art methods including the winner of the Netflix competition are based on non-negative matrix factorization of the partially observed user-rating matrix. The factorized matrix can be used to fill out the unobserved entries in a way similar to latent factor analysis [4, 16, 9, 8].

Each of the above methods focuses exclusively on user ratings. In some cases item information is available (movie genre, actors, directors, etc) which have lead to several approaches that combine voting information with item information e.g., [1, 15, 19].

Our method differs from the methods above in that it constructs a full probabilistic model on preferences,

Kill Bill 1	⇒	Kill Bill 2
Maid in Manhattan	⇒	The Wedding Planner
Two Weeks Notice	⇒	Miss Congeniality
The Royal Tenenbaums	⇒	Lost in Translation
The Royal Tenenbaums	⇒	American Beauty
The Fast and the Furious	⇒	Gone in 60 Seconds
Spider-Man	⇒	Spider-Man 2
Anger Management	⇒	Bruce Almighty
Memento	⇒	Pulp Fiction

Maid in Manhattan	⇒	Pulp Fiction
Maid in Manhattan	⇒	Kill Bill: 1
How to Lose a Guy in 10 Days	⇒	Pulp Fiction
The Royal Tenenbaums	⇒	Pearl Harbor
The Wedding Planner	⇒	The Matrix
Peal Harbor	⇒	Memento
Lost in Translation	⇒	Pearl Harbor
The Day After Tomorrow	⇒	American Beauty
The Wedding Planner	⇒	Raiders of the Lost Ark

Figure 10: Top rules discovered by kernel smoothing estimate on Netflix. Left: like A \Rightarrow like B. Right: like A \Rightarrow dislike B.

it is able to handle heterogeneous preference information (not all users must specify the same number of preference classes) and does not make any parametric assumptions. In contrast to previous approaches it enables not only the prediction of item ratings, but also the discovery of association rules and the estimation of probabilities of interesting events.

6 Summary

Estimating distributions from tied and incomplete data is a central task in many applications with perhaps the most obvious one being collaborative filtering. An accurate estimator \hat{p} enables going beyond the traditional item-rank prediction task. It can be used to compute probabilities of interest, find association rules, and perform a wide range of additional data analysis tasks.

We demonstrate the first non-parametric estimator for such data that is computationally tractable i.e., polynomial rather than exponential in n . The computation is made possible using generating function and dynamic programming techniques.

We examine the behavior of the estimator \hat{p} in three sets of experiments. The first set of experiments involves estimating probabilities of interest such as $p(i \prec j)$. The second set of experiments involves predicting preferences of held-out items which is directly applicable in recommendation systems. In this task, our estimator outperforms other memory based methods (to which it is similar functionally) and performs similarly to state-of-the-art methods that are based on non-negative matrix factorization. In the third set of experiments we examined the usage of the estimator in discovering association rules such as $i \prec j \Rightarrow k \prec l$.

References

- [1] C. Basu, H. Hirsh, and W. Cohen. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the National Conference on Artificial Intelligence*, pages 714–720, 1998.
- [2] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of Uncertainty in Artificial Intelligence*, 1998.
- [3] P. Diaconis. *Group Representations in Probability and Statistics*. Institute of Mathematical Statistics, 1988.
- [4] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.
- [5] David Heckerman, David Maxwell Chickering, Christopher Meek, Robert Rounthwaite, and Carl Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1, 2000.
- [6] J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, page 237. ACM, 1999.

Cluster	Movie Titles
1	American Beauty, Lost in Translation, Pulp Fiction, Kill Bill I,II, Memento, The Royal Tenenbaums, Napoleon Dynamite,...
2	Spider-Man, Spider-Man II
3	Lord of the Rings I,II,III
4	The Bourne Identity, The Bourne Supremacy
5	Shrek, Shrek II
6	Meet the parents, American Pie
7	Indiana Jones and the Last Crusade, Raiders of the Lost Ark
8	The Patriot, Pearl Harbor, Men of Honor, John Q, The General's Daughter, National Treasure, Troy, The Italian Job,...
9	Miss Congeniality, Sweet Home Alabama,Two Weeks Notice,50 First Dates,The Wedding Planner,Maid in Manhattan,Titanic,...
10	Men in Black I,II, Bruce Almighty, Anger Management, Mr. Deeds, Tomb Raider, The Fast and the Furious
11	Independence Day, Con Air, Twister, Armageddon, The Rock, Lethal Weapon 4, The Fugitive, Air Force One

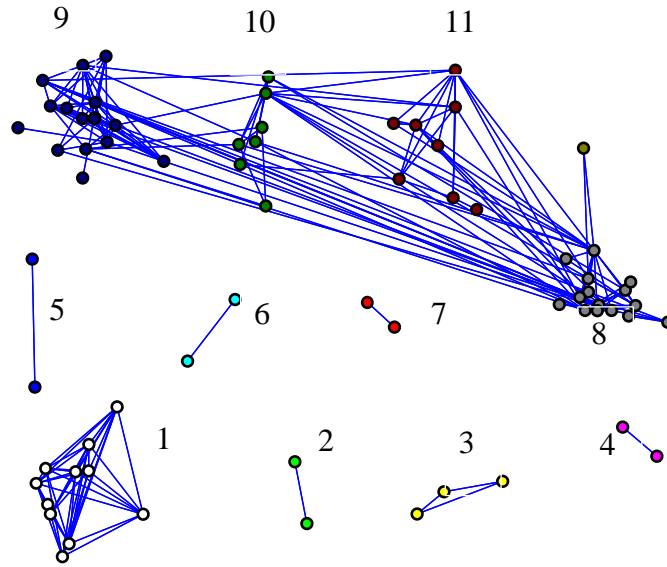


Figure 11: A graph corresponding to the 100 most rated Netflix movies where edges represent high affinity as determined by the rule discovery process (see text for more details).

- [7] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1):115, 2004.
- [8] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 4(1):1–24, 2010.
- [9] N.D. Lawrence and R. Urtasun. Non-linear matrix factorization with gaussian processes. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 601–608. ACM, 2009.
- [10] G. Lebanon and Y. Mao. Non-parametric modeling of partially ranked data. *Journal of Machine Learning Research*, 9:2401–2429, 2008.
- [11] C. L. Mallows. Non-null ranking models. *Biometrika*, 44:114–130, 1957.
- [12] J. I. Marden. *Analyzing and modeling rank data*. CRC Press, 1996.
- [13] B. Marlin. Modeling user rating profiles for collaborative filtering. In *Advances in Neural Information Processing Systems*, 2004.
- [14] D. M. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory- and model-based approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, 2000.
- [15] A. Popescul, L. H. Ungar, D. M. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, 2001.
- [16] J.D.M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*, page 719. ACM, 2005.
- [17] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the Conference on Computer Supported Cooperative Work*, 1994.
- [18] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, 2001.
- [19] A.I. Schein, A. Popescul, L.H. Ungar, and D.M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260. ACM, 2002.
- [20] R. P. Stanley. *Enumerative Combinatorics*, volume 1. Cambridge University Press, 2000.
- [21] L.H. Ungar and D.P. Foster. Clustering methods for collaborative filtering. In *AAAI Workshop on Recommendation Systems*, 1998.
- [22] G.R. Xue, C. Lin, Q. Yang, W.S. Xi, H.J. Zeng, Y. Yu, and Z. Chen. Scalable collaborative filtering using cluster-based smoothing. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 114–121. ACM New York, NY, USA, 2005.